EXHIBIT 1

CLEAN VERSION OF THE ENTIRE SET OF PENDING CLAIMS

1. (three times amended, last amended 7/8/2002) A microprocessor chip, comprising:

instruction pipeline circuitry; and

table lookup circuitry designed to retrieve an entry from a table, each entry of the table being associated with a corresponding address range of an address space translated by address translation circuitry of the microprocessor chip, each entry describing a likelihood of the existence of an alternate coding of instructions located in the respective corresponding address range, the table lookup circuitry operable as part of the basic instruction cycle of executing an instruction of a non-supervisor mode program for execution on the microprocessor chip, the table being stored in storage that is architecturally invisible to programs in the native architecture of at least some instructions executed by the microprocessor chip;

interrupt circuitry cooperatively designed with the instruction pipeline circuitry to trigger a synchronous interrupt on execution of an instruction of a process, wherein the architectural definition of the instruction of the process does not call for an interrupt, a trigger for the interrupt being synchronously based at least in part on the table entry corresponding to the address of the instruction of the process, the interrupt circuitry being designed to invoke a handler for the interrupt, the handler being responsive to a content of the table entry to affect the instruction pipeline circuitry to effect control of an architecturally-visible data manipulation behavior or control transfer behavior of the instruction of the process, based at least in part on the contents of a table entry corresponding to the address range in which the instruction of the process lies.

2. (three times amended, last amended 7/8/2002) A method, comprising the steps of:

as part of the basic instruction cycle of executing an instruction of a non-supervisor mode program executing on a computer, consulting a table, the table having entries that are

indexed by the address within an address space of instructions executed, entries of the table containing attributes of instructions whose addresses index to the respective table entries; and

controlling an architecturally-visible data manipulation behavior or control transfer behavior of the instruction based at least in part on a content of a table entry indexed by the address of the instruction.

61 end

3. (twice amended, last amended 7/8/2002) The method of claim 2, wherein the control of control transfer behavior includes transfer of execution control to a second instruction for execution, the second instruction being an instruction other than an instruction architecturally-defined to be the successor instruction of the instruction.

4. (amended, last amended 8/10/2001) The method of claim 3, wherein the second instruction is coded in an instruction set architecture (ISA) different than the ISA of the executed instruction.

5. (as filed, 10/28/1999) The method of claim 2, wherein the control of architecturally-visible data manipulation behavior includes changing an instruction set architecture under which instructions are interpreted by the computer.

6. (as filed 10/28/1999) The method of claim 2, wherein the behavior control includes selecting between two different instruction set architectures, and the computer includes instruction pipeline circuitry designed to effect interpretation of computer instructions under the two instruction set architectures alternately.

7. (once amended, last amended 8/10/2001) The method of claim 2, wherein:

entries of the table describe a likelihood of the existence of an alternate coding of instructions located in respective corresponding address ranges.

8. (once amended, last amended 8/10/2001) The method of claim 2, wherein:

entries of the table correspond to pages managed by a virtual memory manager, and wherein circuitry for locating an entry of the table is integrated with virtual memory address translation circuitry of the computer.

9. (twice amended, last amended 7/8/2002) The method of claim 2, further comprising the steps of:

triggering a synchronous interrupt on execution of the instruction based at least in part on a memory state of the computer and the address of the instruction, wherein the architectural definition of the instruction in the instruction's native architecture does not call for an interrupt.

~~78~~ 10. (new 7/8/2002) The method of claim 9, wherein a portion of the memory state relevant to the interrupt trigger includes a content of the table entry indexed by the address of the instruction, wherein the architectural definition of the instruction does not call for an interrupt.

~~35~~ 11. (added 1/9/2001) The method of claim 2, wherein the table entry indexed by the address of the instruction is associated with a range of instruction addresses.

~~79~~ 12. (new 7/8/2002) The method of claim 2, wherein:

the controlling of instruction behavior includes altering a manipulation of data or control transfer behavior of the instruction in a manner incompatible with the architectural definition of the instruction in the instruction's native architecture.

~~80~~ 13. (new 7/8/2002) The method of claim 2, wherein:

the step of consulting the table as part of executing an instruction is performed as part of fetching the instruction.

14 ~~10.~~ (twice amended, last amended 7/8/2002) A microprocessor chip, comprising:

instruction pipeline circuitry;

table lookup circuitry designed to index into a table by a memory address of a memory reference arising during execution of an architecturally-defined instruction, and to retrieve a table entry corresponding to the address, the table entry being distinct from the memory referenced by the memory reference;

G6

the instruction pipeline circuitry being responsive to the contents of the table entry to alter a manipulation of data or control transfer behavior of the instruction in a manner incompatible with the architectural definition of the instruction in the instruction's native architecture.

15 ~~11.~~ (three times amended, last amended 7/8/2001) The microprocessor chip of claim 14 ~~10~~, further comprising:

G7

a binary translator programmed to translate at least a selected portion of a computer program from a first binary representation to a second binary representation; and

wherein the pipeline control circuitry is further designed to initiate a determination of whether to transfer control from an execution of the architecturally-defined instruction, the architecturally-defined instruction being an instruction of the first binary representation of the program, to the second binary representation, and effective to initiate the determination with neither a query nor a control transfer to the second binary representation being coded into the first binary representation.

16 ~~12.~~ (twice amended, last amended 7/8/2002) The microprocessor chip of claim 14 ~~10~~, further comprising:

G8

interrupt circuitry cooperatively designed with the instruction pipeline circuitry to trigger a synchronous interrupt on execution of the architecturally-defined instruction based at least in part on the contents of the table entry, wherein the architectural definition of the architecturally-defined instruction in the instruction's native architecture does not call for an interrupt.

17 16

~~13.~~ (twice amended, last amended 7/8/2001) The microprocessor chip of claim ~~12~~, further comprising:

interrupt handler software designed to service the interrupt and to effect the altering of a control transfer behavior, the altering being in the form of returning control from the handler to an instruction flow other than the instruction flow triggering the interrupt, the returned-to instruction flow for carrying on non-error handling normal processing logically equivalent to the architecturally-defined processing of instructions following the altered instruction.

G8 end

14. (amended, last amended 8/10/2001) A microprocessor chip, comprising:

instruction pipeline circuitry;

address translation circuitry; and

a lookup structure having entries associated with corresponding address ranges generated by the instruction pipeline circuitry and translated by the address translation circuitry, the entries describing a likelihood of the existence of an alternate coding of instructions located in the respective corresponding address range.

15. (as filed 10/28/1999) The microprocessor chip of claim 14, wherein the entry is an entry of a translation look-aside buffer.

16. (as filed 10/28/1999) The microprocessor chip of claim 14, wherein the alternate coding is coded in an instruction set architecture (ISA) different than the ISA of the instruction located in the address range.

18

21 ~~17~~. (amended, last amended 7/8/2002) The microprocessor chip of claim ~~14~~, wherein:

G9

the instruction pipeline circuitry is responsive to the contents of the lookup structure entry to affect an architecturally-visible manipulation of data or control transfer behavior of the instruction.

22 ~~18~~. (twice amended, last amended 7/8/2002) The microprocessor chip of claim ~~14~~ 18, further comprising:

interrupt circuitry cooperatively designed with the instruction pipeline circuitry to trigger a synchronous interrupt on execution of an instruction of a process based at least in part on a lookup structure entry associated with the address of the instruction, wherein the architectural definition of the instruction in the instruction's native architecture does not call for an interrupt.

23 ~~19~~. (four times amended, last amended 7/8/2002) A microprocessor chip, comprising:

instruction pipeline circuitry; and

interrupt circuitry cooperatively designed with the instruction pipeline circuitry to trigger a synchronous interrupt on execution of an instruction of a process based at least in part on a memory state of the computer and the address of the instruction, wherein the architectural definition of the instruction in the instruction's native architecture does not call for an interrupt.

20. (as filed, 10/28/1999) The microprocessor chip of claim 19, further comprising:

interrupt handler software designed to service the interrupt and to return control to an instruction flow of the process other than the instruction flow triggering the interrupt, the returned-to instruction flow for carrying on non-error handling normal processing of the process.

21. (as filed, 10/28/1999) The microprocessor chip of claim 20, wherein the interrupt handler software is programmed to change an instruction set architecture under which instructions are interpreted by the computer.

22. (amended, last amended 8/10/2001) The microprocessor chip of claim 20, wherein the returned-to instruction flow is logically equivalent to the instruction flow beginning at the interrupted instruction.

27 23

~~23~~. (amended, last amended 7/8/2002) The microprocessor chip of claim ~~19~~, further comprising:

table lookup circuitry designed to index into a table by a memory address within an address space of a memory reference arising during execution of the instruction, and to retrieve a table entry corresponding to the memory-reference address;

G12

the instruction pipeline circuitry being responsive to the contents of the table entry to affect an architecturally-visible manipulation of data or control transfer behavior of the instruction.

28 23

G13

~~58~~. (amended 7/8/2002) The microprocessor chip of claim ~~19~~, wherein a trigger for the interrupt is further based on a value of the instruction.

29 23

~~59~~. (new 7/8/2002) The microprocessor chip of claim ~~19~~, wherein:

G14

the memory state on which triggering the interrupt is based includes an entry of a table indexed by the address of instructions fetched for execution, entries of the table containing attribute indicia of instructions whose addresses index to the respective entries, the table entries being architecturally invisible to an architecture for execution in the instruction pipeline circuitry.

24. (once amended, last amended 8/10/2001) A method, comprising the steps of:

as an integral part of processing an instruction in instruction pipeline circuitry of a computer, consulting a lookup structure of entries, entries of the lookup structure corresponding to address ranges translated by address translation circuitry, and the entries describe a likelihood of the existence of an alternate coding of instructions located in the respective corresponding address ranges.

30

31 ~~26~~. (twice amended, last amended 7/8/2002) The method of claim ~~24~~, further comprising the step of:

G15

161

G

altering a behavior of the instruction in a manner incompatible with the architectural definition of the instruction in the instruction's native architecture, based at least in part on a content of the lookup structure entry corresponding to the address range containing the instruction.

32 ~~27~~. (twice amended, last amended 7/8/2001) The method of claim 30 ~~24~~,

wherein each lookup structure entry corresponds to a page managed by a virtual memory manager, and wherein circuitry for locating a lookup structure entry is integrated with virtual memory address translation circuitry of the computer.

33 ~~28~~. (twice amended, last amended 7/8/2002) The method of claim 30 ~~24~~, further comprising the step of:

based at least in part on a content of the lookup structure entry, transferring control to the alternative coding, the alternative coding being an instruction flow of the process other than the instruction flow triggering the consulting, the transferred-to instruction flow being programmed to carry on non-error handling normal processing of the process.

34 ~~29~~. (twice amended, last amended 7/8/2002) The method of claim 30 ~~24~~, further comprising the step of:

based at least in part on a content of the lookup structure entry, triggering a synchronous interrupt, wherein the architectural definition of the instruction in the instruction's native architecture does not call for an interrupt.

57. (new 8/10/2001) The method of claim 24, further comprising the step of:

after determining that the existence of an alternate coding is likely, consulting a second table, the entries of the second table definitively indicating entry points for initiating such alternate codings as exist.

36 ~~60~~. (new 7/8/2002) The method of claim 30 ~~24~~, wherein:

the lookup structure entries are architecturally invisible to programs executing in an instruction set architecture executed by the instruction pipeline circuitry.

25. (amended, last amended 8/21/2001) A method, comprising the steps of:

as an integral part of processing an instruction in instruction pipeline circuitry of a computer, consulting a lookup structure of entries, each entry corresponding to an address range translated by address translation circuitry, and describing a likelihood of the existence of an alternate coding of instructions located in the respective corresponding address range; and

as a result of the consulting, changing an instruction set architecture under which instructions are interpreted by the computer.

~~30~~ 38. (twice amended, last amended 7/8/2001) A method, comprising the steps of:

on execution of an instruction of a process in a computer, triggering a synchronous interrupt based at least in part on a memory state of the computer and the address of the instruction, wherein the architectural definition of the instruction in the instruction's native architecture does not call for an interrupt.

31. (new 10/12/2000) The method of claim 30, further comprising the step of:

servicing the interrupt and returning control to an instruction flow of the process other than the instruction flow triggering the interrupt, the returned-to instruction flow being programmed to carry on non-error handling normal processing of the process.

~~32~~ 40. (amended 7/8/2002) The method of claim ~~31~~ 39, further comprising the step of:

changing an instruction set architecture under which instructions are interpreted by the computer in handler software for the interrupt.

33. (new 10/12/2002) The method of claim 32, wherein the instruction text beginning at the returned-to instruction is logically equivalent to the instruction text beginning at the interrupted instruction.

42 34. (twice amended, last amended 7/8/2002) The method of claim 38, further comprising:

indexing into a table by a memory address within an address space of a memory reference arising during execution of an instruction, and to retrieve a table entry corresponding to the address;

responding to a content of the table entry to affect an architecturally-visible manipulation of data or control transfer behavior of the instruction.

43 39. (amended, last amended 7/8/2002) A method, comprising the steps of:

as part of the basic instruction cycle of executing an architecturally-defined instruction of a non-supervisor mode program executing on a computer, retrieving an entry from a table, the table entry being indexed by the address of a memory reference arising during execution of the architecturally-defined instruction, the table entry being distinct from the memory referenced by the memory reference;

based at least in part on a content of the table entry, altering a manipulation of data or control transfer behavior of the architecturally-defined instruction in a manner incompatible with the architectural definition of the architecturally-defined instruction in the architecturally-defined instruction's native architecture.

40. (added 8/10/2002) The method of claim 39, wherein the entries of the table correspond to ranges of memory addresses.

45 41. (once amended, last amended 7/8/2002) The method of claim 43 39, wherein the table entry is indexed by the address within an address space of instructions fetched for execution.

46 42. (once amended, last amended 7/8/2002) The method of claim 43 38, wherein:

entries of the table correspond to respective address ranges, and the table entries describe a likelihood of the existence of an alternate coding of instructions located in the respective corresponding address ranges.

1604

G

47 ~~43~~. (amended, last amended 7/8/2002) The method of claim ~~39~~ 43:

wherein the architectural definition of the architecturally-defined instruction in the architecturally-defined instruction's native architecture does not call for an interrupt;

and further comprising the step of triggering a synchronous interrupt based at least in part on a memory state of the computer and the address of the architecturally-defined instruction.

~~44~~ 48. (amended, last amended 7/8/2002) The method of claim ~~39~~ 43, wherein the control of control transfer behavior includes transfer of execution control to a second instruction for execution.

G ~~20~~ 21 end

~~45~~ 49. (amended, last amended 7/8/2002) The method of claim ~~44~~ 48, wherein the second instruction is coded in an instruction set architecture (ISA) different than the ISA of the architecturally-defined instruction.

---

46. (new, added 8/10/2001) The method of claim 39, wherein the control of architecturally-visible data manipulation behavior includes changing an instruction set architecture under which instructions are interpreted by the computer.

47. (added 8/10/2001) The method of claim 39, wherein the behavior control includes selecting between two different instruction set architectures, and the computer includes instruction pipeline circuitry designed to effect interpretation of computer instructions under the two instruction set architectures alternately.

48. (added 8/10/2001) The method of claim 39:

wherein each entry of the table corresponds to a page managed by a virtual memory manager, and wherein circuitry for locating an entry of the table is integrated with virtual memory address translation circuitry of the computer.

53 50. (amended, last amended 7/8/2002) An apparatus, comprising:

instruction pipeline circuitry; and

table lookup circuitry designed to retrieve a table entry from a table whose entries are indexed by an address within an address space of an instruction fetched for execution by the instruction pipeline circuitry;

the instruction pipeline circuitry being responsive to a content of the table entry to control an architecturally-visible data manipulation behavior or control transfer behavior of the fetched instruction based at least in part on a content of the table entry indexed by the address of the instruction.

G 22

54 53 51. (amended, last amended 7/8/2002) The apparatus of claim 50, wherein the control of control transfer behavior includes transfer of execution control to a second instruction for execution, the second instruction being coded in an instruction set architecture (ISA) different than the ISA of the executed instruction.

55 53 52. (amended, last amended 7/8/2002) The apparatus of claim 50,

wherein entries of the table correspond to pages managed by a virtual memory manager, circuitry for locating a table entry being integrated with virtual memory address translation circuitry of the computer.

---

56 53 53. (new, added 7/8/2002) The apparatus of claim 50, wherein:

G 23

the table entry is stored in storage architecturally invisible to programs executing in the instruction's native architecture.

---

57 54 53 54. (amended, last amended 7/8/2002) The apparatus of claim 50, further comprising:

24 G

circuitry designed to trigger a synchronous interrupt on execution of an instruction of a process, based at least in part on a memory state of the computer and the address of the instruction, wherein the architectural definition of the instruction in the instruction's native architecture does not call for an interrupt.

58 53
55. (amended, last amended 7/8/2002) The apparatus of claim 50, wherein: the table entries are indexed by a virtual address of the instruction.

59 53
56. (amended, last amended 7/8/2002) The apparatus of claim 50, wherein: the table entries are indexed by a physical address of the instruction.

60
61. (new 7/8/2002) A method, comprising the steps of:

as part of the basic instruction cycle of executing an instruction of a non-supervisor mode program fetched for execution on a computer, consulting a table, entries of the table being indexed by addresses of instructions fetched, entries of the table containing attribute indicia of instructions whose addresses index to the respective entries; and

controlling an architecturally-visible data manipulation behavior or control transfer behavior of the fetched instruction based at least in part on a content of a table entry indexed by the address of the fetched instruction, the table entries being architecturally-invisible in the fetched instruction's native architecture.

61 60
62. (new 7/8/2002) The method of claim 61, wherein the control of control transfer behavior includes transfer of execution control to a second instruction for execution, the second instruction being an instruction other than an instruction architecturally-defined to be the successor instruction of the fetched instruction.

62 61
63. (new 7/8/2002) The method of claim 62, wherein the second instruction is coded in an instruction set architecture (ISA) different than the ISA of the fetched instruction.

63 60
64. (new 7/8/2002) The method of claim 61, wherein the control of architecturally-visible data manipulation behavior includes changing an instruction set architecture under which instructions are interpreted by the computer.

64 60
~~65~~. (new 7/8/2002) The method of claim ~~61~~, wherein the behavior control includes selecting between two different instruction set architectures, and the computer includes instruction pipeline circuitry designed to effect interpretation of computer instructions under the two instruction set architectures alternately.

65 60
~~66~~. (new 7/8/2002) The method of claim ~~61~~, wherein:

the attribute indicia describe a likelihood of the existence of an alternate coding of instructions located in respective address ranges corresponding to the table entries.

66 60
~~67~~. (new 7/8/2002) The method of claim ~~61~~:

wherein the architectural definition of the fetched instruction in the fetched instruction's native architecture does not call for an interrupt;

G25 and further comprising the step of triggering a synchronous interrupt based at least in part on a memory state of the computer and the address of the fetched instruction.

67 60
~~68~~. (new 7/8/2002) The method of claim ~~61~~, further comprising the step of:

altering a behavior of the fetched instruction in a manner incompatible with the architectural definition of the fetched instruction in the fetched instruction's native architecture, based at least in part on a content of the table entry indexed by the address of the fetched instruction.

68 60
~~69~~. (new 7/8/2002) The method of claim ~~61~~, wherein:

entries of the table correspond to pages managed by a virtual memory manager, and wherein circuitry for locating an entry of the table is integrated with virtual memory address translation circuitry of the computer.

69
~~70~~. (new 7/8/2002) An apparatus, comprising:

instruction pipeline circuitry; and

table lookup circuitry designed to retrieve a table entry from a table whose entries are indexed by an address of an instruction fetched for execution, the table being stored in

storage that is architecturally invisible to programs in the fetched instruction's native architecture;

the instruction pipeline circuitry being responsive to a content of the table entry to control an architecturally-visible data manipulation behavior or control transfer behavior of the fetched instruction based at least in part on a content of the table entry associated with the address of the fetched instruction.

70 ~~71~~. (new 7/8/2002) The apparatus of claim 69 ~~70~~, wherein the control of control transfer behavior includes transfer of execution control to a second instruction for execution, the second instruction being coded in an instruction set architecture (ISA) different than the ISA of the fetched instruction.

71 ~~72~~. (new 7/8/2002) The apparatus of claim 69 ~~70~~,

G29 wherein entries of the table correspond to pages managed by a virtual memory manager, the table lookup circuitry being integrated with virtual memory address translation circuitry.

72 ~~73~~. (new 7/8/2002) The apparatus of claim 69 ~~70~~, wherein:

entries of the table describe a likelihood of the existence of an alternate coding of instructions located in address ranges corresponding to respective table entries.

73 ~~74~~. (new 7/8/2002) The apparatus of claim 69 ~~70~~, wherein:

the control of the fetched instruction's behavior includes altering a manipulation of data or control transfer behavior of the fetched instruction in a manner incompatible with the architectural definition of the fetched instruction in the fetched instruction's native architecture.

74 ~~75~~. (new 7/8/2002) The apparatus of claim 69 ~~70~~, further comprising the steps of:

triggering a synchronous interrupt on fetch or execution of the fetched instruction, based at least in part on a memory state of the computer and the address of the fetched

instruction, wherein the architectural definition of the fetched instruction in the fetched instruction's native architecture does not call for an interrupt.

75 69
~~76~~. (new 7/8/2002) The apparatus of claim ~~70~~, wherein:
the table entries are indexed by virtual addresses of instructions.

G2 end

76 69
~~77~~. (new 7/8/2002) The apparatus of claim ~~70~~, wherein:
the table entries are indexed by physical addresses of instructions.